Search vs. Decision for Election Manipulation Problems

Edith Hemaspaandra
RIT

Lane A. Hemaspaandra
University of Rochester

Curtis Menton
University of Rochester

Our Modest, Mild, Oh-so-Teentsie Claim:

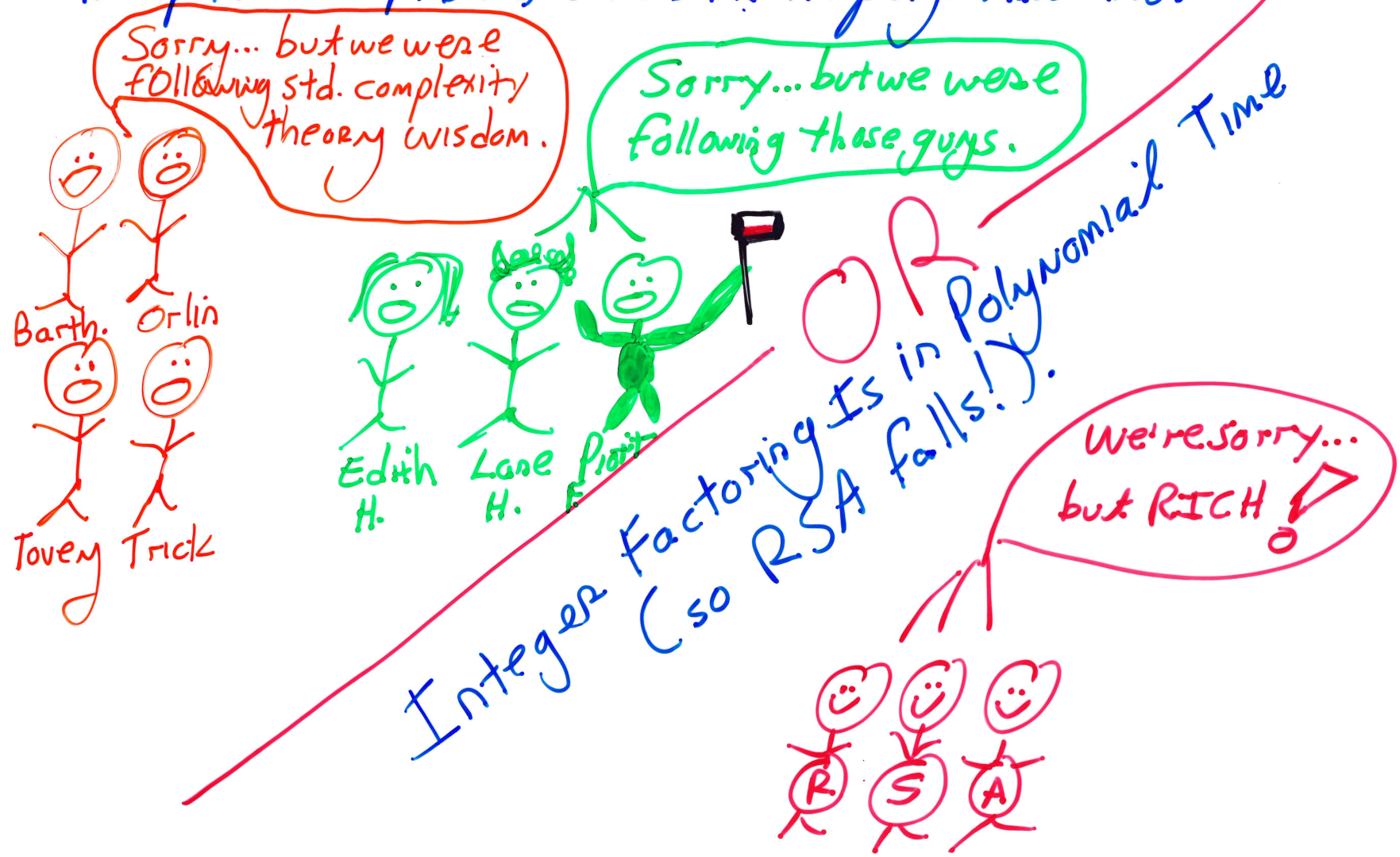
COMSOC's 20-year-standing manipulative-action definitions are flawed

OR

Modern Cryptography's 30-year-standing foundations are flawed

Theorem

There are election systems (even having poly-time winner problems) that under the standard definitions have poly-time manipulation (resp., bribery, control) algorithms, yet one cannot manipulate (resp., bribe, control) them in polynomial time.



If Integer Factoring Is Not Easy, We Have: 3

P:

We can manipulate (bribe, control) this instance so our candidate Eilonwy wins!

Great! What is the manipulation (bribery, control) action that works?

I have no idea! But I know an action exists that works!

So what should we do?

Uh... change the definitions? .... No! Never!!

Darcy, a Decision-Problem loving dragon

Sam, a Search-problem loving Sorcerer

The Key Issue

- Manipulation [BTT89a, BO91], Control [BTT92], and Bribery [FHH09] all have their complexity defined in terms of decision problems, not in terms of search problems.
- Yet any actual attacking agent will want not just an existential claim, but also an ACTION to take!

So... why were these defined in terms of decision problems?

Blame Latvia!

$DTIME[f(n)] \stackrel{?}{\subseteq} NEXP$

$P \subseteq NTIME[f(n)]$

- Manip./Control/Bribery complexity were defined that way because decision problems are how standard complexity theory rolls... at least usually... Every 1st-year grad course: Instance:— Question:—

  e.g.
  SAT is a decision problem: $\{F \mid F \text{ is satisfiable}\}$.

- But, usually, no harm, no foul:
  SAT's decision and search versions are poly-time Turing equivalent!!
  (and the same holds in general for many problems, including all NP-complete problems.... that's why decision problems rule the world (of first-year grad. school and far beyond)....
  except...

Secret Society

P_1

- Down with decision problems!
- We worship the sacred flame of search!
- Because it is a more refined notion: $\equiv_T^P$ smears too much. P_2
- Because it is what matters!

NPMV

NPSV

N.P.A.C.C. JELLYBEANS

FNP/TFNP/PPAD

Standard COMSOC Manipulative-Action Definitions

- E-manipulation (two key types, e.g., unweighted coalition manip:)

[BTT] Given: Candidate set C, nonmanipulative voterset V_1, manipulative voterset V_2, $p \in C$.
Question: Is there some choice of prefs for V_2 so p is a winner in the E-election over C with voters $V_1 \cup V_2$?

- E-bribery (many types, including:)

[FHH] Given: (C, V), $p \in C$, integer $b \geq 0$.
Question: Does there exist a collection of at most b voters, and a way of setting their votes, so that p is a winner under E in the election with all voters voting? (Also: prices/budget.)

- E-control (many types, including "deleting voters":)

[BTT] Given: (C, V), $p \in C$, integer $k \geq 0$.
Question: Is there a $W \subseteq V$, $\|W\| \leq k$, so p is a winner under E in the election $(C, V-W)$.

Manipulative-Action Defs. Part 2

- DESTRUCTIVE versions of manipulation [CSLO07], control [HHR07], and bribery [FHH09]:

  Can p be prevented from being a winner?

- Control: Add/delete/partition of voters/candidates.

  (Partition: TP to match nonunique winner model.)

P_1

- Seeking to model various attacks.
- MANY control types.... but 2 fewer than before this paper:

Theorem:
- DC-RPC-TP = DC-PC-TP.
- DC-RPC-TE = DC-PC-TE.

$P_{1.5}$

Thm:
- DC-RPC-TP-UniqWinner ≠ DC-PC-TP-UniqWinner.
- DC-RPC-TE-UniqWinner = DC-PC-TE-UniqWinner.

P_2

Using observations from inside a proof in [FHHR09]

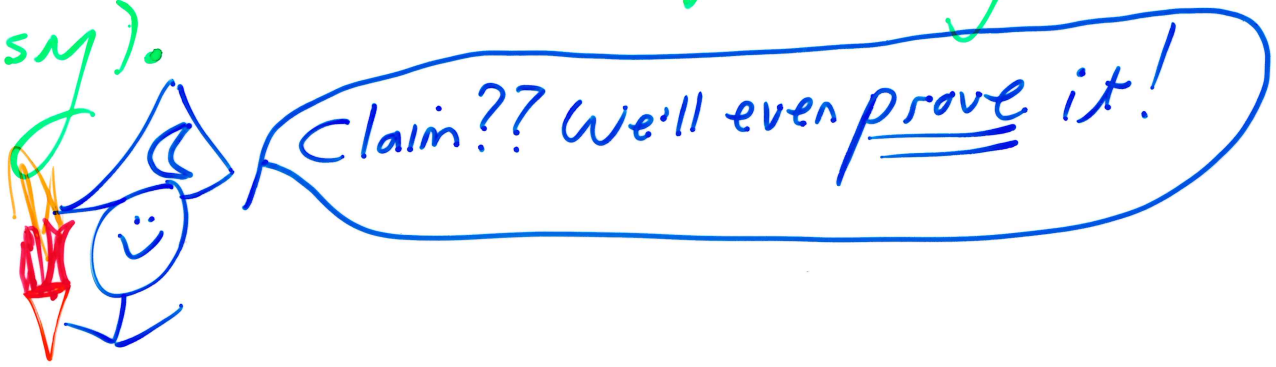
smoking gun

Now, in reality...

- These defs. have worked pretty well so far.

- Existing worries are typically that "easy to attack" is UNDERinclusively defined (due to heuristic/typical-case algorithms/approaches; WARNING: complexity theory limits that argument line's reach - see [FHH "nearly-SP"]).

- We claim the defs. of "easy to attack" are OVERinclusive (unless integer factoring is easy).



Natural Search Versions

"success is impossible"
OR
"Here is the ACTION (manip., bribe, control) that will achieve success."

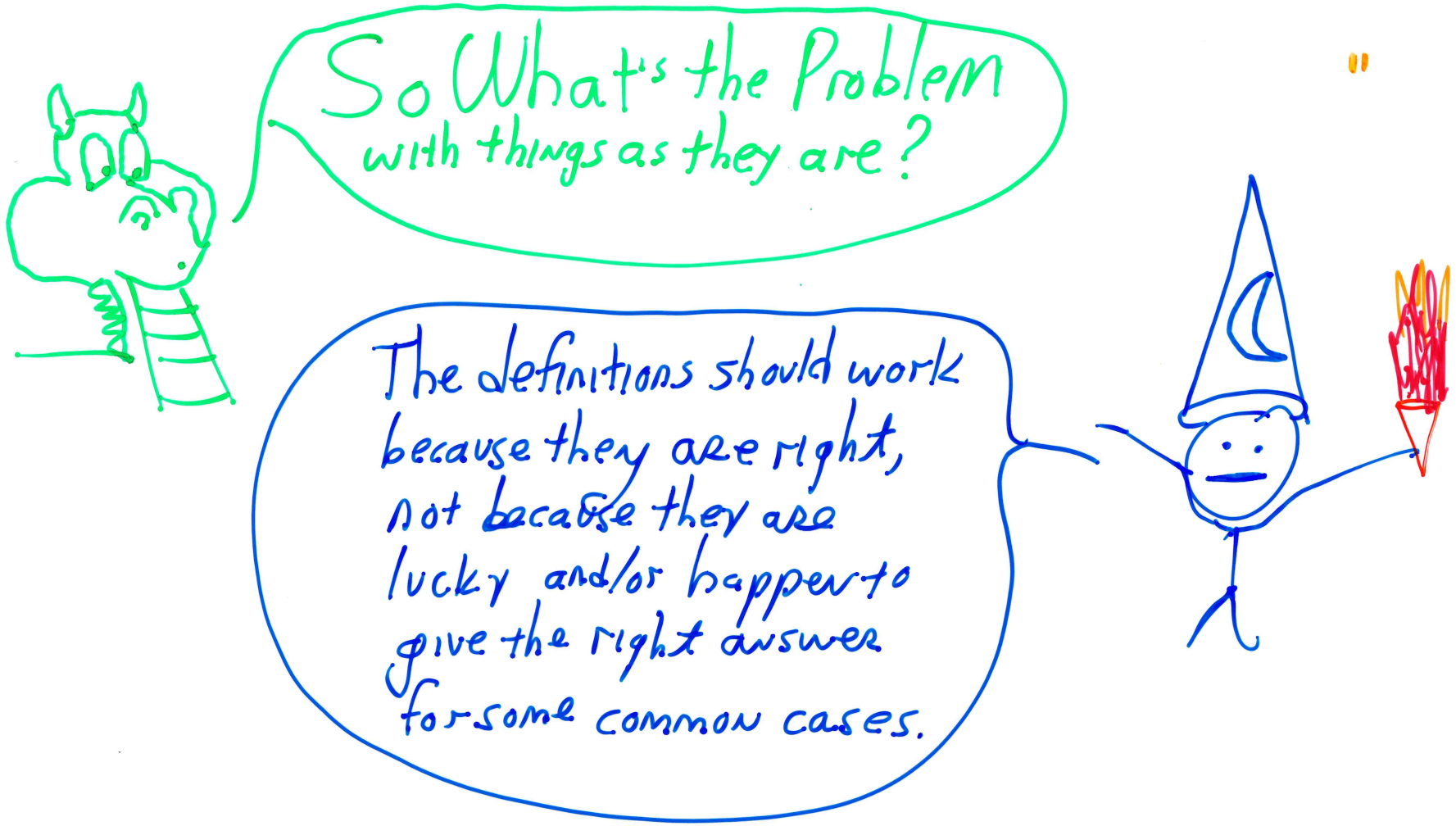
P

Holy Grail?

Good news: BTT/CSL/PR/FHH/...:
We've already got one!!

Since existing papers routinely prove P for decision by giving a P-time algorithm for the search case.

In the lit. "certifiably vulnerable" ([HHRO7]6 also other papers since).



Plus, exactly because we usually look to poly-time search algs. to establish p-time decision, we may be extra-blind to finding cases where they separate.

For Goodness's Sake, Get to the Theorem!

MAIN THM. If integer factoring is hard (or even if $NP \cap coNP \neq P$), then for basically all types of manip. and bribery, and just under ½ the std. types of control, there is an election system $\mathcal{E}$, with a poly-time winner problem, for which the attack type is in poly-time (under the std. decision def.), but its search version is not in polynomial time. (So search does not reduce to decision.)

COMPANION THM. For the Remaining std. control types, search always reduces to decision. (So the above type of result cannot hold for them.)

RESULTS TABLE

	Action	Constructive	Destructive
	Bribery	• $S \not\subseteq D$	• $S \not\subseteq D$
CONTROL BY	Adding Voters	• $S \subseteq D$	• $S \subseteq D$
	Deleting Voters	• $S \subseteq D$	• $S \subseteq D$
	Voter Partition, TP	• $S \not\subseteq D$	• $S \not\subseteq D$
	Voter Partition, TE	• $S \not\subseteq D$	• $S \not\subseteq D$
	Adding Candidates	• $S \subseteq D$	• $S \subseteq D$
	Deleting Candidates	• $S \subseteq D$	• $S \subseteq D$
	Cand. Partition, TP	• $S \not\subseteq D$	• $S \subseteq D$
	Cand. Partition, TE	• $S \not\subseteq D$	• $S \subseteq D$
	Cand. Runoff Partition, TP	• $S \not\subseteq D$	• $S \subseteq D$
	Cand. Runoff Partition, TE	• $S \not\subseteq D$	• $S \subseteq D$
	Unlimited Adding Candidates	• $S \subseteq D$	• $S \subseteq D$
	Manipulation	• $S \not\subseteq D$	• $S \not\subseteq D$

Legend: "$S \subseteq D$": $\forall E$, search poly-time Turing-reduces to decision.
"$S \not\subseteq D$": If $P \neq NP \cap coNP$, $\exists E$ with p-time winner problem with p-time decision version but with search not in p.

} for the given attack type

How is that proven?

(See our ~38-page TR...)

Very carefully! Briefly, we join with the construction of elections a lovely, little-known result from a 1976 tech report, the Borodin-Demers Theorem...

B-D Thm [BD76, Val76]: If $P \neq NP \cap coNP$ then $(\exists A)[A \subseteq SAT \wedge A \in P \wedge (\exists f \in FP)(\exists F \in A)[F(f(F)) \text{ is false}]]$.

So does B-D show that $S \not\subseteq D$ for SAT?

No way! $S \subseteq D$ for SAT!!!! But.....

BD and our constructions yield $S \not\subseteq D$ for many election attacks unless factoring is easy.

By the way, a key hurdle is keeping the decision versions in P. DANGER, WILL ROBINSON: If we are not very careful the decision prob. will ask if an obviously satisfiable formula has a satisfying assignment MEETING EXTRA CONSTRAINTS — fatal!!

Worry #1

So your table shows that $P \neq NP \cap coNP$ is sufficient to get 12 $S \not\subseteq D$ conclusions.

But maybe these are 12 distinct issues, since your table isn't a complete characterization?!

No! We prove that all 12 are disguised versions of the same question in a certain sense: They all stand or fall together!

Σ_1

Why? We completely characterize each in terms of the RHS of the B-D Theorem, by tightly connecting elections to that RHS.

Σ_2

Worry #2 Part 1

In your "S≠D" results, maybe the "hard" search version is only rarely hard, and heuristics can often do well?

We prove "hardness inheritance" results: Our problems are about as often hard as the most often hard NP∩coNP sets! To heuristically conquer our sets, you must also conquer all of NP∩coNP.

Worry #2 Part 2: the formal statement

Theorem: If f is any nondecreasing function, and for some set $A \in NP \cap coNP$ it holds that every p-time membership-in-A algorithm errs, at infinitely many lengths n (resp., at almost every length n), on at least $f(n)$ of the strings up to that length, then for each manipulative action (that appears in our $P \neq NP \cap coNP$ theorems) there will exist an $\in ?O$ and an election system having a poly-time winner problem such that each search algorithm for that manipulative action will err, at infinitely many lengths n (resp., at almost every length n), on at least $f(n^{\in})$ of the strings up to that length, but the decision problem will be in P.

Example If some $NP \cap coNP$ set draws $2^{n^{\Omega(1)}}$ errors from each poly-time alg., so does our search problem.

SUMMARY

- Contributions:
  - If $\Sigma_2^P \neq$ NP ∩ coNP then .... $S \not\leq D$. (For bribery, manip., & many types of control.) ★
  - Completely characterized, identically, our 12 results of that form—all stand or fall together.
  - Showed $S \leq D$ for all other std. control types.
  - Collapsed 2 pairs of destruc. control types: 11 → 9 (11 → 10 in Uniq. Winner).
  - Proved hardness inheritance results: If any NP ∩ coNP set is often hard, then so are our 12 cases — basically at least as often hard.
  - So, perhaps the core defs. should simply be about Σ_2^P the simplicity of the search version.

I'm liking my nice, new, pointy hat!